APPENDIX J

EXPRESS MAIL LABEL
NO.: EL386270487US

Introduction

Scope

This appendix describes the processing of “balls” under the SIP in the ICP machine.

A “Ball” is a circular single component; collections of “balls” can be found in many applications.

In the ICP machine, “balls” processing is only done in pre-defined “balls” areas.

Such processing aims to test the components found in those areas against

design rules and some other pre-defined constraints, as will be discussed in details later.

Glossary

The following is a short list of terms and abbreviations used in the present appendix:

A “connected component” is a chain of CELs (Contour ELements), which can be

either open or closed. In principle, each vertex of such a chain holds information on the CEL

connecting this vertex to its neighbour along the chain.

A “generalized circle” is a structure defined over a single connected component.

A “generalized circle” holds information on the extent by which this circle can be considered a “good” one, based on both design rules and other pre-defined constraints.

The information stored in the generalized circle structure is such that a full and accurate report

on all defects detected on that component, can be derived from it.

An “isolated circle” is a circle which was found to be "good". See section “Circles Sampling” for the definition of a “good” circle suggested by the implementation described here.

“Balls” Configuration Requirements

SIP

The ICP/SIP implementation of balls learning procedure expects as input a list of descriptions

of radius ranges. A range is defined by *min* and *max* values, within which a *nominal* radius value resides (the nominal radius value may or may not be provided as input).

In addition, some relevant test data is attached to each range. For example, to test circularity, a parameter relevant to the tolerance on fitness to circle is typically used.

Application

To go along with application needs, the specific format of data typically used by the SIP can be provided by the SIP Server, from various application formats. Note: the application typically only deals with diameter values, which are typically converted into radius values by the SIP Server.

The following combinations of parameter formats may be supported:

i. A list of *nominals*, cannot be null (derived from Application parameter: *balls_nominal_diameter*, conversion by multiplication with 0.5).

ii. A list of pairs (*min, max)* defining each range, can be null (derived from Application parameter: *balls_min_max_nominal_diameter*, conversion by multiplication with 0.5*)*.

iii. A list, or a single value of *percentage_radius_deviation* or *absolute_radius_deviation* (derived from Application parameters: *balls_percentage_diameter_deviation,* for which convertion is done by multiplication with 2.0, and *balls_absolute_diameter_deviation* - for which conversion is done by multiplication with 0.5).

iv. *adjust_to_nominal_radius* parameter (Expert parameter: *balls_adjust_to_nominal*), applicable in learning scenarios only, default is FALSE. If TRUE, radiuses of circles to be learned are adjusted to the closest matching nominal (if such exist).

v. A list, or a single value of test-data parameters:

Note: if both *abs* and *percentage* parameters are given, the stricter value of the two is

taken as the parameter.

- *maximum_percentage_nick_depth / maximum_absolute_nick_depth,* and *maximum_percentage_protrusion_depth / maximum_absolute_protrusion_depth* .
- *maximum_absolute_defect_area / maximum_percentage_defect_area.*
- *maximum_absolute_defect_length / maximum_percentage_defect_length.*
- *circle_fit_sensitivity* and *percentage_circle_fit..*

Note: All the above, but the last two parameters (which are Expert parameters: *balls_circle_fit_sensitivity,* and *balls_percentage_circle_fit*) - are Application parameters. No conversion diameter-to-radius is typically used for none of the above.

A list of *nominals* is preferably always provided. Providing a list of pairs (min, max) is optional.

If only *nominals* are given, at least one of the parameters defined above, is preferably provided. In that case, *min* and *max* values will be computed as follows:

min = nominal - absolute_radius_deviation, or,
min = nominal * (1 - *percentage_radius_deviation).*

max = nominal + absolute_radius_deviation, or,
max = nominal * (1 + *percentage_radius_deviation).*

In the above calculations, if both parameters *(absolute_radius_deviation* and *percentage_radius_deviation)* exist, the smaller value of the two is taken as the *min*, and the higher value of the two is taken as the *max*.

If a list of pairs *(min, max)* exist, the above calculation will serve to refine the boundaries of those ranges, only if the parameter *allow_range_adjust* is set to TRUE.

"Balls" Learning Scheme

Circles Sampling

Description of the "balls" learning procedure: The goal of this procedure is to process connected components found in a given "balls" area, and to produce all ball or circle information that can be derived from each component in that area. Note that this procedure processes information retrieved from one camera (slice) at a time.

• *Input*: A connected component.

• *Output*: A generalized circle.

• *Process*:

1. The (camera coordinates) points defining a connected component are served to compute a circle parameters: center and radius (possibly in aligned coordinates).

2. Each point is examined against a robust estimator of the circle parameters, and the "best" points are identified (this is done using an internal parameter *circle_fit_tolerance*, on which the user has no control). Those "best" points are used to refine the circle estimated parameters.

3. At this point, a circle undergoes a test for a match to nominal radius range. The circle's estimated radius is examined against the available radius range list; A match to range *i* is set if the circle's radius resides within the *i*'s range boundaries (*min*, *max*).

4. To complete the sampling procedure, some indicators of the quality of the estimation, and/or some other tests can be calculated and examined against internal tolerances. In accordance with a preferred embodiment of the present invention, a fitness to circle test is applied. The parameters *circle_fit_sensitivity* and *percentage_circle_fit* are taken from the radius range table, where the estimated radius is served as the access key. If the estimated radius does not match any radius range in the table, default values are used for the parameters.

A "good" circle is thus a connected component for which all test results are above given thresholds, and whose radius matches one and only one* radius range.

* A match to more than one radius is a miss-definition of the radius ranges. A warning on this will be given at loading, and on circle sampling (execution).

No defect will be reported, and the circle will be learned.

Analysis of Unified Circle Information

The above procedure is applied to any "balls" window. At the final processing stage, all "balls" windows are processed together to eliminate duplicate appearances of circles in cameras' overlap regions (matching of two circles in this case is based on matching their centers up to *max_alignment_shift* tolerance, taken from the corresponding Expert parameter, and on matching their radiuses up to some *radius_deviation* tolerance, which is an internal parameter).

The circles are classified into two groups: "good" and "bad". Each such group undergoes a different handling and testing.

"Good" Circles Processing:

First, duplicate appearances of same "good" circles are removed, then each "good" circle undergoes the following tests:

Area reduction test.

The Area reduction test verifies that the area of the connected component underlying that circle is approximately the same as the expected area of the circle itself. If there is a reduction in the circle's area which is above the *defect_area* parameter taken from the relevant entry in the radius_range table, a SHAPE_AREA_REDUCTION sip_defect is reported.

- Nick-protrusion test.

The Nick-protrusion test examines the points on the contour of the circular component. A preferred nick-protrusion test is described in Applicant's copending Israel application 131092.

Those points are supposed to reside at a distance equals to the radius of the circle from the circle's center. A Nick is reported if a point on the contour is located at distance smaller than (radius - inner_threshold) from the circle center. A protrusion is reported if a point of the contour is located at distance larger than (radius + outer_threshold) from the circle center. The parameters *inner_threshold and* *outer_threshold* are taken from the entry in radius range table corresponding to the circle radius. They are derived from the Application parameters defining the criteria for nicks and protrusions, respectively.

Each Nick/Protrusion point is reported as NICK_PROT sip_defect.

- Clean area test.

This test verifies that the points on the contour of the circular component do not corrupt the "clean area" defined by a circle of radius *clean_radius* around the center of the sampled circle. At the moment, the parameter *clean_radius* is set to the value of *min_radius* of the radius range corresponding to the sampled circle radius. A point on the contour which violates the clean area criterion is reported as NICK_PROT_ON_BALL sip_defect.

If *adjust_to_nominal_radius* is TRUE, the radius of each learned circle is set to the matching nominal radius. As mentioned before, the adjustment is applicable only if nominal radiuses were defined.

The reference circles are stored in aligned coordinates.

"Bad" Components Processing:

Defected components which are instances of "good" circle, resulting from camera slicing are removed. Other defective components are tested according to their polarity.

- Same polarity as panel.

The length of the bad component is tested against *max_exposed_metal_length* threshold.

(The length of a connected component is defined as the largest diagonal of its bounding box.) If length exceeds the threshold, a SHAPE_LENGTH_EXCEEDS_THRESHOLD sip_defect is reported, at the point which is the center of that bounding box.

- Inverse Polarity: treat the bad circle as Pinhole.

To test the component as pinhole, we first find the circle within which this component resides. We use the radius of that circle to get thresholds from radius range table, and

test length of the pinhole against defect_length parameter, and its area against defect_area parameter. If one of those tests proves this pinhole to be above threshold, it is

reported as SHAPE_LENGTH_EXCEEDS_THRESHOLD, or SHAPE_AREA_EXCEEDS_THRESHOLD, and its center is reported as PINHOLE.

Inspection Scenario

In principle, the inspection scenario for circles is similar to the learning scenario. A delicate detail is the treatment of radius ranges; As in the learning scenario, a list of radius ranges is used, and the radius of an approximated online circle serves to access the list of radius ranges. Note that the radius range list provided can be the same list of radius ranges that was used during learning, or a new range list can be provided. If a totally new range list is used within which no radius of any reference circle resides, defects may encounter later on Excess-Missing tests. These test (see details below) are performed in "balls" inspection areas to verify that the appearance of circles on the panel matches the appearance of reference circles.

The inspection processing of connected components in "balls" window is the following:

• *Input*: A connected component.

• *Output*: Defect reports.

• *Process*:

1. Compute a (possibly aligned) circle (center and radius) out of the online connected component.
2. Identify the "best" points, and refine the circle estimated parameters.
3. Examine the circle's radius against the available radius range list.
4. Calculate the quality of the estimation (or other tests), and examine it against given tolerances.

5. After unifying circle information from all slices, classify into two groups and test "good" and "bad" circles as in the learning scenario.

6. Perform extra tests specific to inspection scenario:

The goal of the two following tests is to verify that any reference circle has an online circle matching it exactly (position and radius in our case), and vice versa. Matching in this context means that there is a reference circle with the same radius (up to the internal *radius_deviation* tolerance) and at the same center location (up to the Expert parameter *max_registration_shift* tolerance) as the online circle.

•

• Excess Test

This process prevents the appearance of unexpected circles, that is, connected components with circular shape, whose radius resides within the range of possible radii, but for which no matching reference circle exist.

• Missing Test

This process verifies that there is no reference circle which does not have any matching online circle. Any appearance of online circle matching a reference circle is marked. The number of appearances serves to identify those for which no match was found. Overmatching is also reported.

Extra "Balls" Services

"Balls" Statistics

Statistical information is useful for estimating the parameters typically used at learning (Design- Rule test). Therefore, statistical information is provided at the end of a learning scenario by a special task named *balls_measurenments*. The information is presented in two forms: an histogram representing the distribution of radius values *(a)* in the given radius ranges, and *(b)* in sub-ranges of fixed pre-defined size. Thus, the user can apply an initial learning process with only one pair of *min* and *max* values representing the range (0, Inf), then use the resulting histogram of radius distribution in fixed sub-ranges over that range, to determine an exact setting of the radius range list.

In addition, information on all circles that were learned (before being adjusted to nominals, if such option is used) are outputted; This includes the circles center and radius, and its fit - rounded to the closest fit level. The fit levels typically used are determined by two parameters given to this task: *minimum_circle_fit* and *n_circle_fit_levels;* the task will accordingly output *n_circle_fit_levels* fit levels, from *minimum_circle_fit* and up.

Update "Balls" reference data

"Balls" reference data can be updated according to information retrieved from the application. The goal of this procedure is to allow the user to confirm the learning process. The user has the ability to select reference circles to be taken out from the reference, simply by providing

a list of points which are close enough to the centers of the circles to be removed (those points can be, e.g., provided by the application from 'mouse clicks' on that circle).

The update procedure produces a new reference data (in SIP format) out of the given reference data and the list of points of removal.

Defect information

Table 1:

DEFECT TYPE	DEFECT DESCRIPTION
EXCESS_CELS = 1	Deviation of points residing on the contour of a connected component (cels) from a circular contour.
SHAPE_MISMATCH = 10	Online shape exists but cannot be matched t any reference circle.
SHAPE_EXCESS = 12	Excess circle with the same polarity as the panel
SHAPE_MISSING = 13	Reference circle was expected at this point but was not found.
SHAPE_OVERMATCH = 14	More than one online circles match a single reference circle at this point.
PINHOLE = 15	Excess shape with polarity opposite to panel.
SHAPE_LENGTH_EXCEEDS_THRESHOLD	Excess shape with length (long diagonal axis) that exceeds threshold.

= 16	
SHAPE_AREA_EXCEEDS_THRESHOLD = 17	Excess shape with area that exceeds threshold.
SHAPE_AREA_REDUCTION = 18	Circle for which a massive reduction (over a threshold) in area has occurred.
OPEN_SHAPE = 19	A non-closed shape.
NO_RADIUS_MATCH = 50	A connected component for which no matching radius was found in the radius-range table.
NO_FIT_TO_CIRCLE = 51	A connected component which does not fit to a circle.
NICK_PROT = 40	A point of nick or protrusion (located on a circular contour).
NICK_PROT_ON_BALL = 41	A point of nick or protrusion inside clean area of a circle.
MISSING_DATA_FOR_COMPUTATION = 201	No reference circles found in :balls" window, or Failure in creating connected components for that window, or No radius range test data was found.
UNKNOWN = 250	Unknown circle defect.